1) 회사가 레거시 애플리케이션을 Amazon EC2로 마이그레이션 중입니다. 이 애플리케이션에서는 소스 코드에 저장된 사용자 이름과 암호를 사용하여 MySQL 데이터베이스에 연결합니다. 이 데이터베이스를 Amazon RDS for MySQL DB 인스턴스로 마이그레이션할 계획입니다. 회사는 마이그레이션 프로세스의 일부로 데이터베이스 자격 증명을 저장하고 자동으로 교체하는 안전한 방법을 구현하려고 합니다.

이러한 요구 사항을 충족하는 방법은 무엇입니까?

A) 데이터베이스 자격 증명을 Amazon Machine Image (AMI)의 환경 변수에 저장합니다. AMI를 대체하여 자격 증명을 교체합니다.
B) 데이터베이스 자격 증명을 AWS Systems Manager 파라미터 스토어에 저장합니다. 파라미터 스토어를 구성하여 자격 증명을 자동으로 교체합니다.
C) 데이터베이스 자격 증명을 EC2 인스턴스의 환경 변수에 저장합니다. EC2 인스턴스를 다시 시작하여 자격 증명을 교체합니다.
D) 데이터베이스 자격 증명을 AWS Secrets Manager에 저장합니다. Secrets Manager를 구성하여 자격 증명을 자동으로 교체합니다.

2) 개발자는 사용자가 의견을 게시하고 실시간에 가까운 피드백을 받을수 있는 웹 애플리케이션을 설계하고 있습니다.

어떤 아키텍처가 이러한 요구 사항을 충족합니까? (2개를 선택하십시오.)

A) AWS AppSync 스키마와 해당 API를 생성합니다. Amazon DynamoDB 테이블을 데이터 스토어로 사용합니다.
B) Amazon API Gateway에서 WebSocket API를 생성합니다. AWS Lambda 함수를 백엔드로 사용하고 Amazon DynamoDB 테이블을 데이터 스토어로 사용합니다.
C) Amazon RDS 데이터베이스에서 지원하는 AWS Elastic Beanstalk 애플리케이션을 생성합니다. 수명이 긴 TCP/IP 소켓을 허용하도록 애플리케이션을 구성합니다.
D) Amazon API Gateway에서 GraphQL 엔드포인트를 생성합니다. Amazon DynamoDB 테이블을 데이터 스토어로 사용합니다.
E) Amazon CloudFront에서 WebSocket을 활성화합니다. AWS Lambda 함수를 오리진으로 사용하고 Amazon Aurora DB 클러스터를 데이터 스토어로 사용합니다.

3) 개발자가 애플리케이션에 가입 및 로그인 기능을 추가 중입니다. 이 애플리케이션에서는 고객 사용자 지정 분석 솔루션에 API를 호출하여 사용자 로그인 이벤트를 기록해야 합니다.

개발자는 이러한 요구 사항을 충족하기 위해 어떤 작업 조합을 수행해야 합니까? (2개를 선택하십시오.)

A) Amazon Cognito를 사용해 가입 및 로그인 기능을 제공합니다.
B) AWS IAM을 사용해 가입 및 로그인 기능을 제공합니다.
C) AWS Config 규칙을 구성해 사후 인증 이벤트로 트리거되는 API 호출을 생성합니다.
D) Amazon API Gateway 메서드를 호출해 사후 인증 이벤트로 트리거되는 API 호출을 생성합니다.
E) AWS Lambda 함수를 실행해 사후 인증 이벤트로 트리거되는 API 호출을 생성합니다.

4) 회사가 AWS 계정에서 REST API를 위해 Amazon API Gateway를 사용 중입니다. 보안 팀은 다른 한 AWS 계정의 IAM 사용자만 API에 액세스할 수 있도록 하려고 합니다.

보안 팀은 이러한 요구 사항을 충족하기 위해 어떤 작업 조합을 수행해야 합니까? (2개를 선택하십시오.)

A) IAM 허가 정책을 생성해 각 IAM 사용자에 연결합니다. API 메서드 권한 부여 유형을 AWS_IAM으로 설정합니다. 서명 버전 4를 사용해 API 요청에 서명합니다.
B) Amazon Cognito 사용자 풀을 생성하고 풀에 각 IAM 사용자를 추가합니다. API의 메서드 권한 부여 유형을 COGNITO_USER_POOLS로 설정합니다. Amazon Cognito에서 IAM 자격 증명을 사용하여 인증하고 ID 토큰을 요청 헤더에 추가합니다.
C) Amazon Cognito 자격 증명 풀을 생성하고 풀에 각 IAM 사용자를 추가합니다. API의 메서드 권한 부여 유형을 COGNITO_USER_POOLS로 설정합니다. Amazon Cognito에서 IAM 자격 증명을 사용하여 인증하고 액세스 토큰을 요청 헤더에 추가합니다.
D) 각 IAM 사용자만 액세스를 허용하도록 API의 리소스 정책을 생성합니다.
E) 각 IAM 사용자만 액세스를 허용하도록 API의 Amazon Cognito 권한 부여자를 생성합니다. API의 메서드 권한 부여 유형을 COGNITO_USER_POOLS로 설정합니다.

5) 개발자가 텍스트 파일을 .pdf 파일로 변환하는 애플리케이션을 빌드 중입니다. 텍스트 파일은 별도의 애플리케이션에 의해 소스 Amazon S3 버킷에 기록됩니다. 개발자는 파일이 Amazon S3에 도착했을 때 이 파일을 읽은 후 AWS Lambda를 사용하여 .pdf 파일로 변환하려고 합니다. 개발자는 Amazon S3 및 Amazon CloudWatch Logs에 액세스를 허용하는 IAM 정책을 작성했습니다.

개발자는 Lambda 함수가 올바른 권한을 가지고 있는지 확인하기 위해 어떤 작업을 수행해야 합니까?

A) AWS IAM을 사용하여 Lambda 실행 역할을 생성합니다. 이 역할에 IAM 정책을 연결합니다. Lambda 실행 역할을 Lambda 함수에 할당합니다.
B) AWS IAM을 사용하여 Lambda 실행 사용자를 생성합니다. 이 사용자에 IAM 정책을 연결합니다. Lambda 실행 사용자를 Lambda 함수에 할당합니다.
C) AWS IAM을 사용하여 Lambda 실행 역할을 생성합니다. 이 역할에 IAM 정책을 연결합니다. Lambda 함수에서 IAM 역할을 환경 변수로 저장합니다.
D) AWS IAM을 사용하여 Lambda 실행 사용자를 생성합니다. 이 사용자에 IAM 정책을 연결합니다. Lambda 함수로 IAM 사용자 자격 증명을 환경 변수로 저장합니다.

6) 회사가 여러 지리적 위치에 AWS 워크로드가 있습니다. 개발자는 us-west-1 리전에서 Amazon Aurora 데이터베이스를 생성했습니다. 이 데이터베이스는 고객이 관리하는 AWS KMS 키를 사용하여 암호화됩니다. 개발자는 이제 이와 동일하게 암호화된 데이터베이스를 us-east-1 리전에서 생성하려고 합니다.

이 작업을 수행하기 위해서는 개발자가 어떤 접근 방식을 취해야 합니까?

A) us-west-1 리전에서 데이터베이스의 스냅샷을 생성합니다. 이 스냅샷을 us-east-1 리전으로 복사하고 us-east-1 리전에서 KMS 키를 지정합니다. 복사된 스냅샷에서 데이터베이스를 복원합니다.
B) us-west-1 리전에서 데이터베이스의 암호화되지 않은 스냅샷을 생성합니다. 이 스냅샷을 us-east 1 리전에 복사합니다. 복사된 스냅샷에서 데이터베이스를 복원하고 us-east-1 리전에서 KMS 키를 사용하여 암호화를 활성화합니다.
C) 데이터베이스의 암호화를 비활성화합니다. us-west-1 리전에서 데이터베이스의 스냅샷을 생성합니다. 이 스냅샷을 us-east 1 리전에 복사합니다. 복사된 스냅샷에서 데이터베이스를 복원합니다.
D) us-east-1 리전에서 us-west-1 리전 데이터베이스의 최신 자동 백업을 복원하도록 선택합니다. us-east 1 리전에서 KMS 키를 사용하여 암호화를 활성화합니다.

7) 개발자가 데이터베이스의 로드를 줄이고 성능을 높이기 위해 회사의 기존 레코드 스토리지 애플리케이션에 Amazon ElastiCache for Memcached를 추가 중입니다. 이 개발자는 일반적인 레코드 처리 패턴 분석에 따라 지연 로딩을 사용하기로 결정했습니다.

다음 중 지연 로딩을 올바르게 구현할 수 있는 의사 코드(Pseudo-code)의 예는 무엇입니까?

A)
```
record_value = db.query("UPDATE Records SET Details = {1} WHERE ID == {0}", record_key, record_value)
cache.set (record_key, record_value)
```

B)
```
record_value = cache.get(record_key)
if (record_value == NULL)
    record_value = db.query("SELECT Details FROM Records WHERE ID == {0}", record_key)
    cache.set (record_key, record_value)
```

C)
```
record_value = cache.get (record_key)
db.query("UPDATE Records SET Details = {1} WHERE ID == {0}", record_key, record_value)
```

D)
```
record_value = db.query("SELECT Details FROM Records WHERE ID == {0}", record_key)
if (record_value != NULL)
    cache.set (record_key, record_value)
```

8) 개발자가 Amazon EC2 인스턴스 플릿에서 실행되는 애플리케이션의 성능을 추적하려고 합니다. 이 개발자는 평균 및 최대 요청 지연 시간과 같은 플릿 전체의 통계를 본 후 추적하려고 합니다. 개발자는 평균 응답 시간이 임계값을 초과하면 즉시 알림을 받고자 합니다.

이러한 요구 사항을 충족하는 솔루션은 무엇입니까?

A) 각 인스턴스에서 cron 작업을 구성해 응답 시간을 측정하고 Amazon S3 버킷에 저장된 로그 파일을 1분마다 업데이트합니다. 로그 파일을 읽고 Amazon Elasticsearch Service(Amazon ES) 클러스터에 새 항목을 쓰는 AWS Lambda 함수를 트리거하는 Amazon S3 이벤트 알림을 사용합니다. Kibana 대시보드에 결과를 시각화합니다. 응답 시간이 임계값을 초과하면 Amazon SNS 주제에 알림을 전송하도록 Amazon ES를 구성합니다.

B) 응답 시간을 시스템 로그에 쓰도록 애플리케이션을 구성합니다. Amazon Inspector 에이전트를 설치한 후 지속적으로 로그를 읽고 응답 시간을 Amazon EventBridge에 전송하도록 구성합니다. EventBridge 콘솔에서 지표 그래프를 봅니다. 평균 응답 시간 측정치가 임계값을 초과하면 Amazon SNS 알림을 전송하도록 EventBridge 사용자 지정 규칙을 구성합니다.

C) 응답 시간을 로그에 쓰도록 애플리케이션을 구성합니다. 애플리케이션 로그를 CloudWatch Logs로 스트리밍하도록 인스턴스에 Amazon CloudWatch 에이전트를 설치하고 구성합니다. 로그에서 응답 시간의 지표 필터를 생성합니다. CloudWatch 콘솔에서 지표 그래프를 봅니다. 평균 응답 시간 측정치가 임계값을 초과하면 Amazon SNS 알림을 전송하도록 CloudWatch 경보를 생성합니다.

D) 인스턴스에 AWS Systems Manager Agent를 설치한 후 응답 시간을 모니터링하고 이를 사용자 지정 지표로서 Amazon CloudWatch로 전송하도록 구성합니다. Amazon QuickSight에서 지표 그래프를 봅니다. 평균 응답 시간 측정치가 임계값을 초과하면 Amazon SNS 알림을 전송하도록 CloudWatch 경보를 생성합니다.

9) 개발자가 로컬에서 애플리케이션을 테스트 중이며 이를 AWS Lambda에 배포했습니다. 패키지 크기를 제한 이하로 유지하기 위해 배포 파일에 종속성이 포함되지 않았습니다. 원격으로 애플리케이션을 테스트할 때, 종속성 누락 때문에 함수가 실행되지 않습니다.

다음 중 어떤 접근 방법으로 이 문제를 해결할 수 있습니까?

A) Lambda 콘솔 에디터를 사용하여 코드를 업데이트하고 누락된 종속성을 포함합니다.

B) 누락된 종속성이 있는 .zip 파일을 추가로 생성하고 이 파일을 원본 Lambda 배포 패키지에 포함합니다.

C) Lambda 함수의 환경 변수에서 누락된 종속성에 대한 참조를 추가합니다.

D) 누락된 종속성을 포함하는 Lambda 함수에 계층을 연결합니다.

10) 개발자가 Amazon API Gateway를 사용하는 웹 애플리케이션을 빌드 중입니다. 개발자는 개발 및 운영(dev 및 prod) 워크로드에 서로 다른 환경을 유지하려고 합니다. API는 하나의 AWS Lambda 함수에서 별칭 2개를 사용하여 지원됩니다. 하나는 dev이며 다른 하나는 prod입니다.

최소한의 구성으로 이를 달성하려면 어떻게 해야 합니까?

A) 각 환경에 REST API를 생성하고 API를 해당 Lambda 함수의 별칭 dev 및 prod와 통합합니다. 그런 다음 각 단계에 두 API를 배포하고 단계 URL을 사용하여 이 API에 액세스합니다.

B) 하나의 REST API를 생성하고 별칭 자리에 단계 변수를 사용하여 이를 Lambda 함수와 통합합니다. 그런 다음 API를 두 단계(dev 및 prod)에 배포하고 단계별로 다른 별칭을 값으로 사용하여 단계 변수를 생성합니다. 다른 스테이지 URL을 사용하여 API에 액세스합니다.

C) 하나의 REST API를 생성하고 이를 Lambda 함수의 dev 별칭과 통합하여 dev 환경에 배포합니다. canary가 Lambda prod 별칭과 통합되는 prod용 canary 릴리스 배포를 구성합니다.

D) 하나의 REST API를 생성하고 이를 Lambda 함수의 prod 별칭과 통합하여 prod 환경에 배포합니다. canary가 Lambda dev 별칭과 통합되는 dev용 canary 릴리스 배포를 구성합니다.

답

1) D – AWS Secrets Manager는 데이터베이스, 애플리케이션, 서비스 및 다른 IT 리소스에 액세스하는 데 필요한 자격 증명을 보호하는 데 효과적입니다. 이 서비스를 사용하면 수명 주기가 끝날 때까지 데이터베이스 자격 증명, API 키 및 기타 보안 암호를 손쉽게 교체, 관리, 검색할 수 있습니다. 사용자 및 애플리케이션이 Secrets Manager API를 호출해 보안 암호를 검색하기 때문에 민감한 정보를 일반 텍스트 형태로 하드 코딩할 필요가 없습니다. Secrets Manager를 사용하면 보안 암호 교체가 가능하며, Amazon RDS, Amazon Redshift 및 Amazon DocumentDB 통합 기능이 기본 제공됩니다.

2) A, B – AWS AppSync를 사용하면 사용자가 유연한 API를 생성하여 데이터 소스가 하나 이상인 데이터에 안전하게 액세스하고 이를 조작 및 결합할 수 있기 때문에 애플리케이션 개발이 간소화됩니다. AWS AppSync는 애플리케이션에서 필요한 데이터를 손쉽고 정확하게 얻을 수 있도록 GraphQL을 사용하는 관리형 서비스입니다. AWS AppSync 사용자는 Amazon DynamoDB를 포함한 다양한 데이터 소스에서 실시간 업데이트가 필요한 애플리케이션을 포함해 확장 가능한 애플리케이션을 빌드할 수 있습니다. 사용자는 Amazon API Gateway에서 AWS Lambda나 DynamoDB 같은 AWS 서비스 또는 HTTP 엔드포인트의 상태 저장 프론트엔드로 WebSocket API를 생성할 수 있습니다. WebSocket API는 클라이언트 애플리케이션에서 수신한 메시지의 내용을 기반으로 백엔드를 호출합니다. WebSocket API는 요청을 받고 응답하는 REST API와 달리, 클라이언트 애플리케이션과 백엔드 간의 양방향 통신을 지원합니다.

3) A, E – Amazon Cognito를 사용하면 웹 및 모바일 애플리케이션에 사용자 가입, 로그인 및 액세스 제어 기능을 빠르고 손쉽게 추가할 수 있습니다. 또한 사용자는 AWS Lambda 함수를 생성하여 사용자 지정 분석 솔루션에 API 호출을 생성하고 이러한 함수를 Amazon Cognito 사전 인증 트리거로 트리거할 수 있습니다.

4) A, D – 리소스 정책을 사용하면 서명 버전 4(SigV4) 프로토콜을 통해 한 AWS 계정의 API 액세스 권한을 다른 AWS 계정의 사용자에게 부여할 수 있습니다.

5) A – AWS Lambda 함수의 실행 역할은 AWS 서비스 및 리소스에 대한 액세스 권한을 부여합니다. 함수를 생성할 때 사용자가 이 역할을 제공하게 되며, 함수가 호출될 때는 Lambda가 이 역할을 담당합니다.

6) A – 사용자가 암호화된 스냅샷을 복사할 경우 스냅샷의 사본도 암호화해야 합니다. 사용자가 리전 간에 암호화된 스냅샷을 복사하는 경우, 소스 스냅샷에 사용한 것과 동일한 AWS KMS 암호화 키를 사용할 수 없습니다. 이는 KMS 키가 리전별로 다르기 때문입니다. 대신에 사용자는 대상 리전에 유효한 KMS 키를 지정해야 합니다.

7) B – 지연 로딩은 필요할 때까지 레코드 로딩을 지연시킨다는 개념입니다. 지연 로딩에서는 먼저 캐시를 확인합니다. 캐시에 레코드가 없는 경우 지연 로딩은 데이터베이스에서 레코드를 검색한 후 이 레코드를 캐시에 저장합니다.

8) C – Amazon CloudWatch 에이전트를 구성하면 로그 및 지표를 CloudWatch로 스트리밍할 수 있습니다. 지표 필터는 CloudWatch Logs에 저장된 로그로 생성할 수 있습니다.

9) D – 사용자는 AWS Lambda 함수를 구성하여 추가 코드와 콘텐츠를 계층의 형태로 가져올 수 있습니다. 계층은 라이브러리, 사용자 지정 실행 시간 또는 기타 종속성을 포함하는 .zip 아카이브입니다. 사용자는 배포 패키지에 라이브러리를 포함할 필요 없이 계층을 통해 함수에서 라이브러리를 사용할 수 있습니다.

10) B – 사용자는 Amazon API Gateway의 배포 단계에서 알파, 베타, 프로덕션 등 각 API의 여러 릴리스 단계를 관리할 수 있습니다. 단계 변수를 사용하여 API 배포 단계가 다양한 백엔드 엔드포인트와 상호 작용하도록 구성할 수 있습니다. 사용자는 API Gateway 단계 변수를 사용하여 여러 버전 및 별칭이 있는 단일 AWS Lambda 함수를 참조할 수 있습니다.